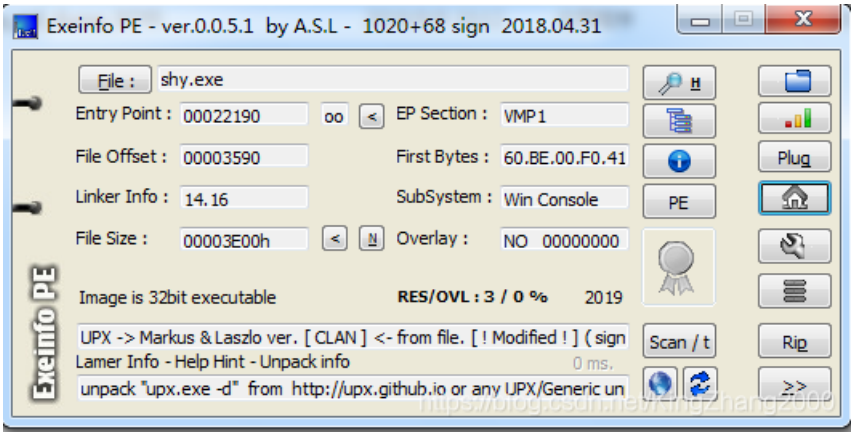
CTF 逆向题 shy

原创

KingZhang2000 于 2019-09-09 11:25:36 发布 1831 收藏 6
分类专栏： CTF 文章标签： CTF 逆向 upx脱壳

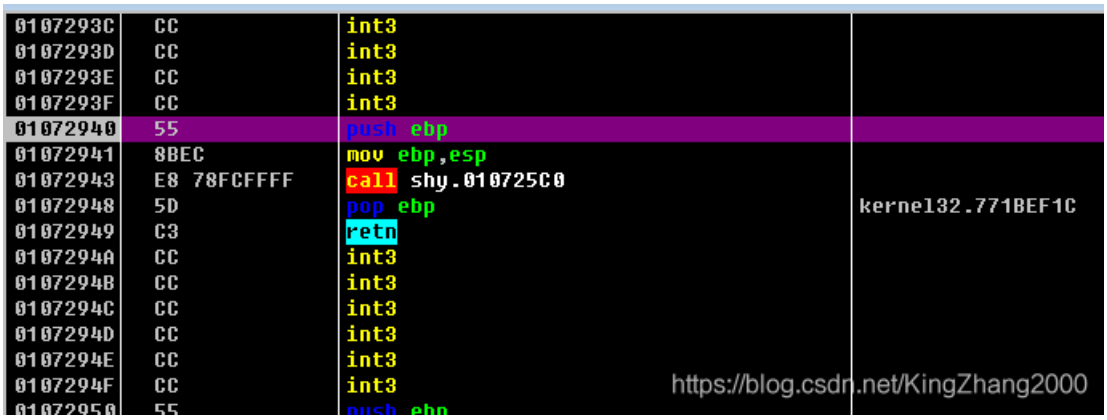
这个题目是攻防大赛的逆向题，是upx的壳，由于当时手头的工具无法脱壳，所以这题也就跳过了。经过学习了手工脱壳后，感觉可以拿这个题练练手，顺便写一个writeup。

首先查壳，丢到ExeinfoPE里面看一下，确定是upx壳

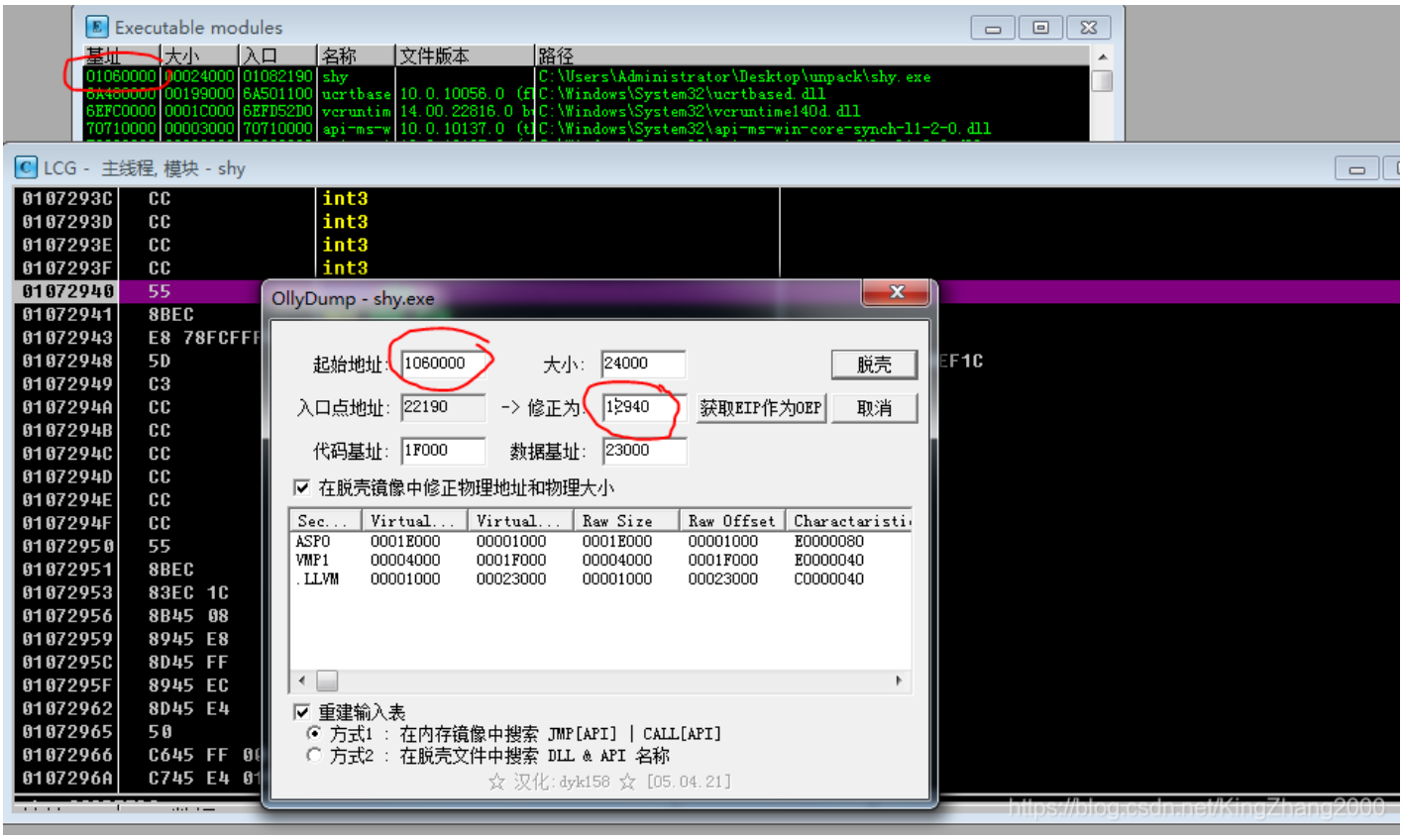


于是丢到OD里面进行脱壳处理，由于是压缩壳，跟踪起来比较麻烦，我选择了个偷懒的办法，下一个api访问断点

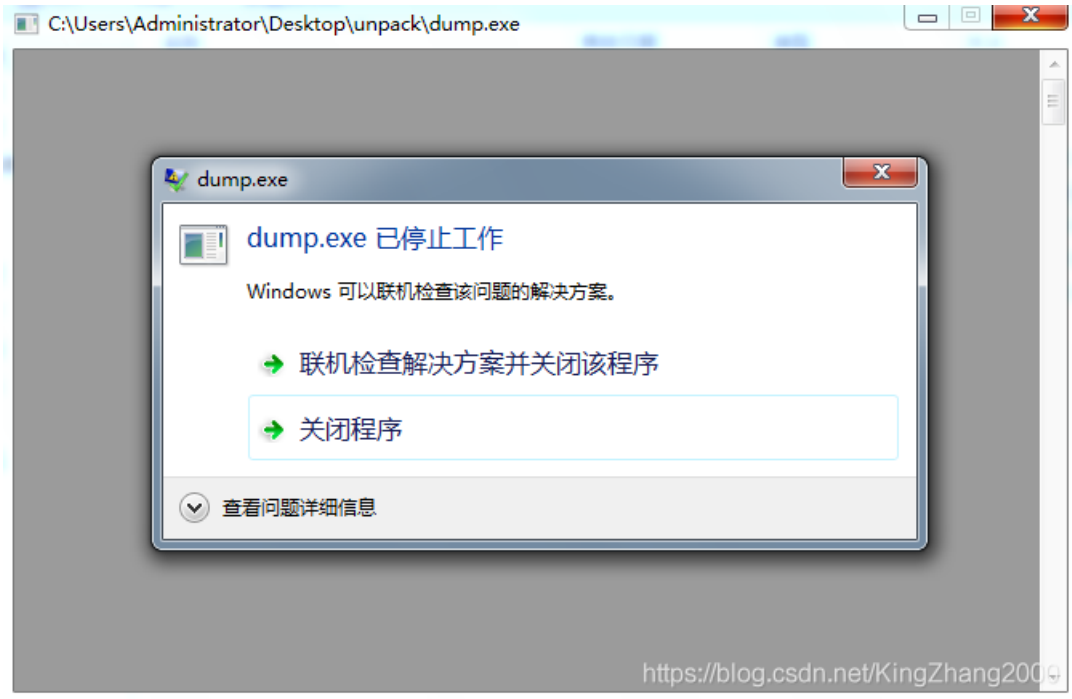
即：VirtualProtect ，运行3次F9后就跑飞了，于是在2次运行后，单步跟踪，到OEP



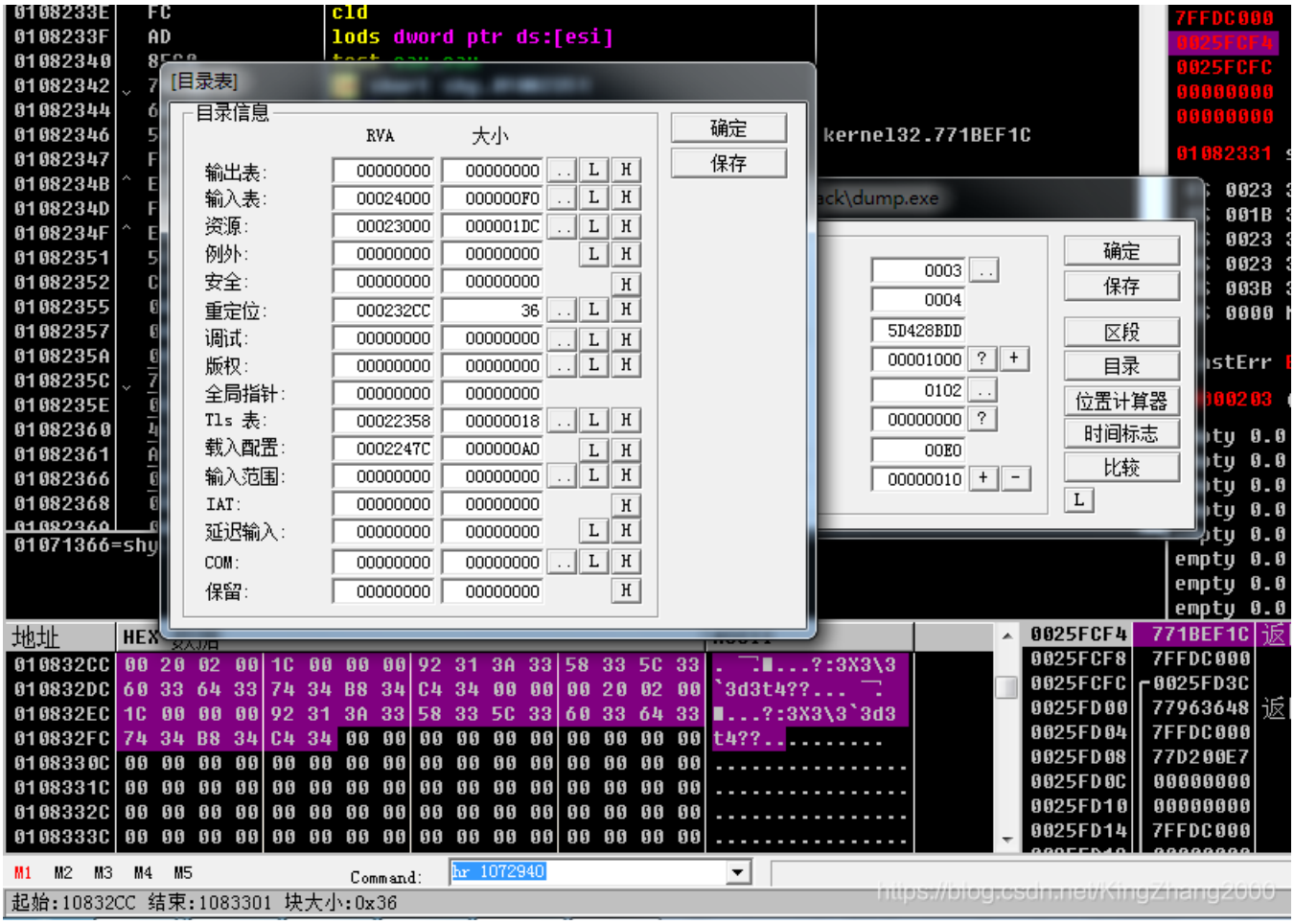
使用OD自带的插件进行脱壳，注意基址和OEP的关系，计算好后填入



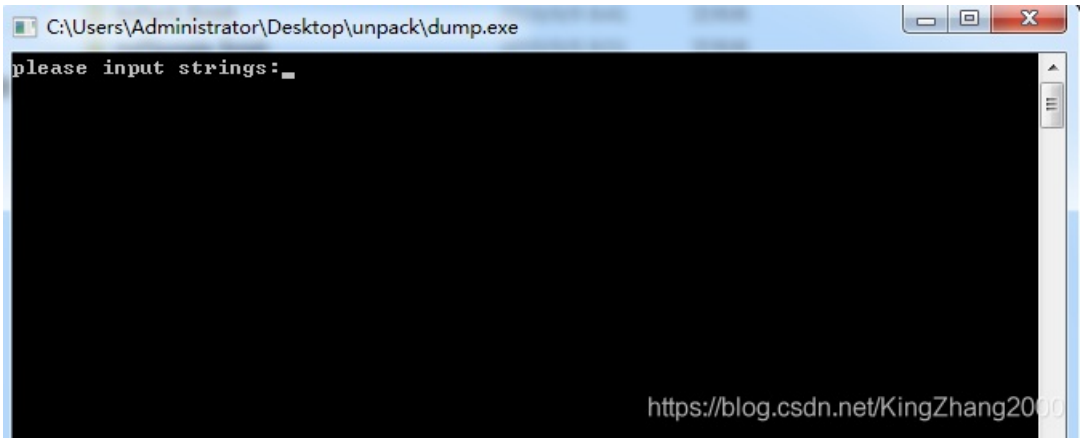
然后点击脱壳，双击发现不能运行。



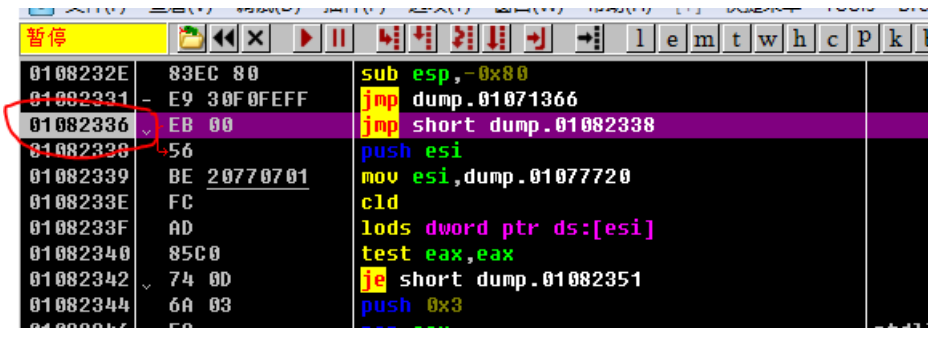
这个问题估计是重定向造成，于是修复一下重定向表的数据。



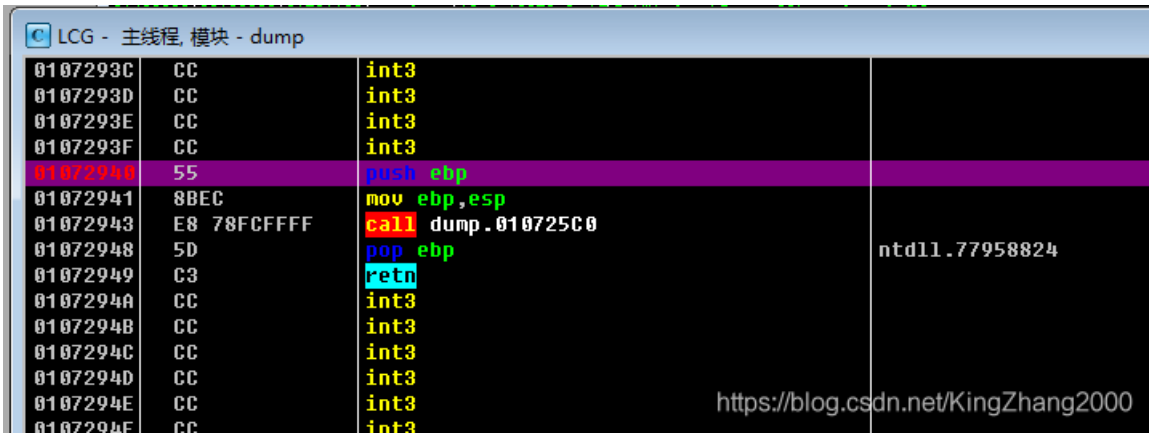
保存后，再次运行可以正常跑起来了



再次用OD载入，发现入口地址不是OEP

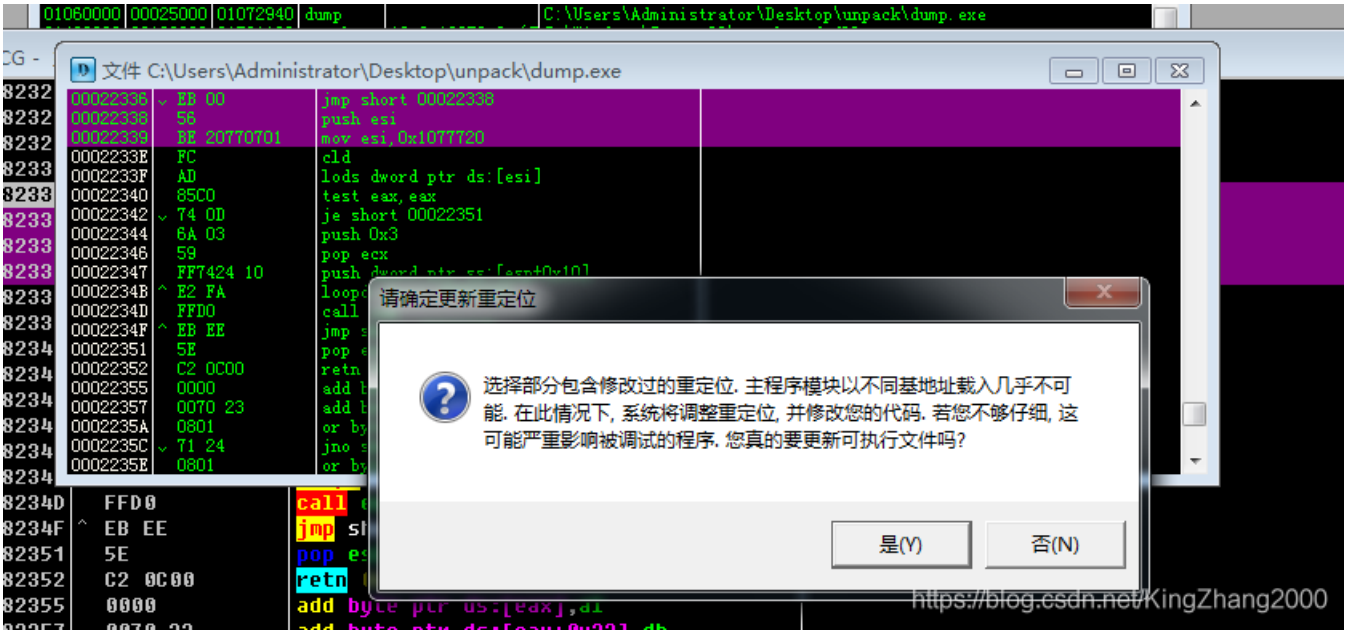


按说是已经解密完成了，于是我定位到OEP （0x1072940）一看究竟。



确实已经解密，那么为了方便调试，我直接用OD改一下入口代码就可以实现了。

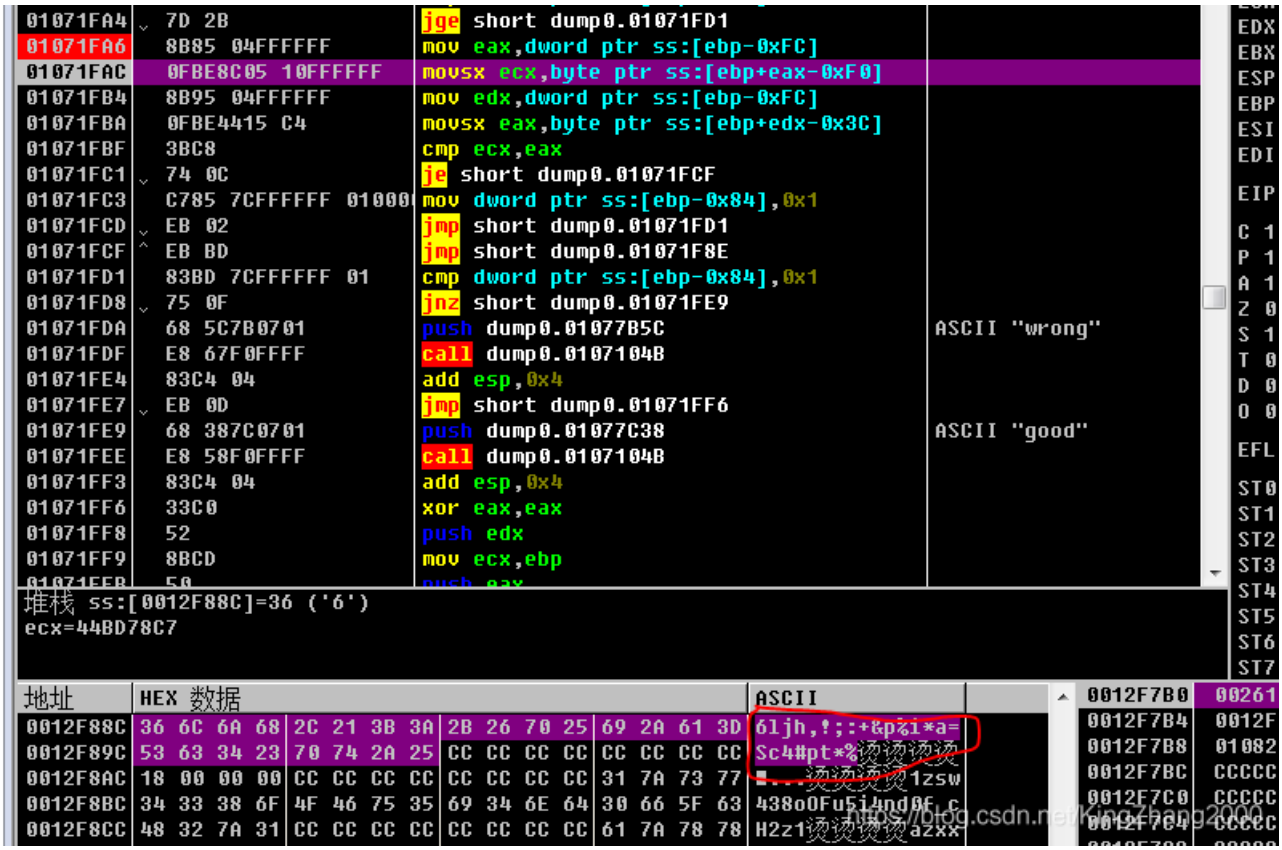
然后把修改完毕的程序，重新保存到文件，由于有重定位会有下图的提示，点击 是 ，然后右键保存一份dump0.exe



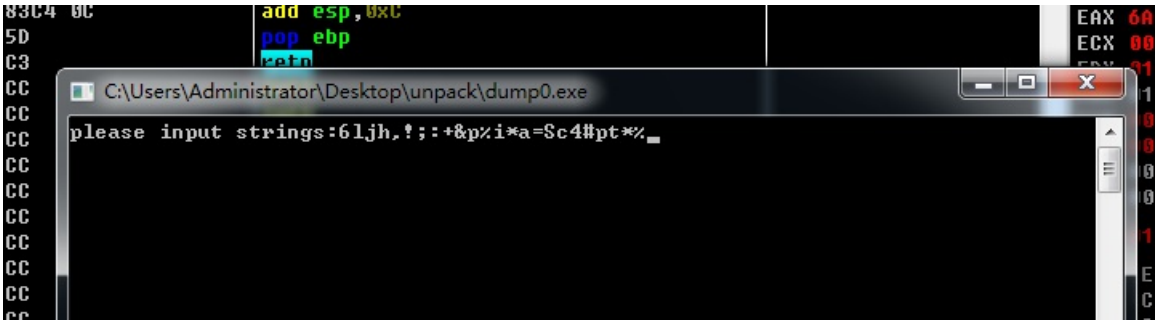
继续，OD载入dump0.exe 发现修改成功，可以直接跳转到OEP行，然后单步跟踪，到输入后，发现后面的代码是个加密处理的代码，于是丢到IDA里面看一下这块对应的反编译代码。

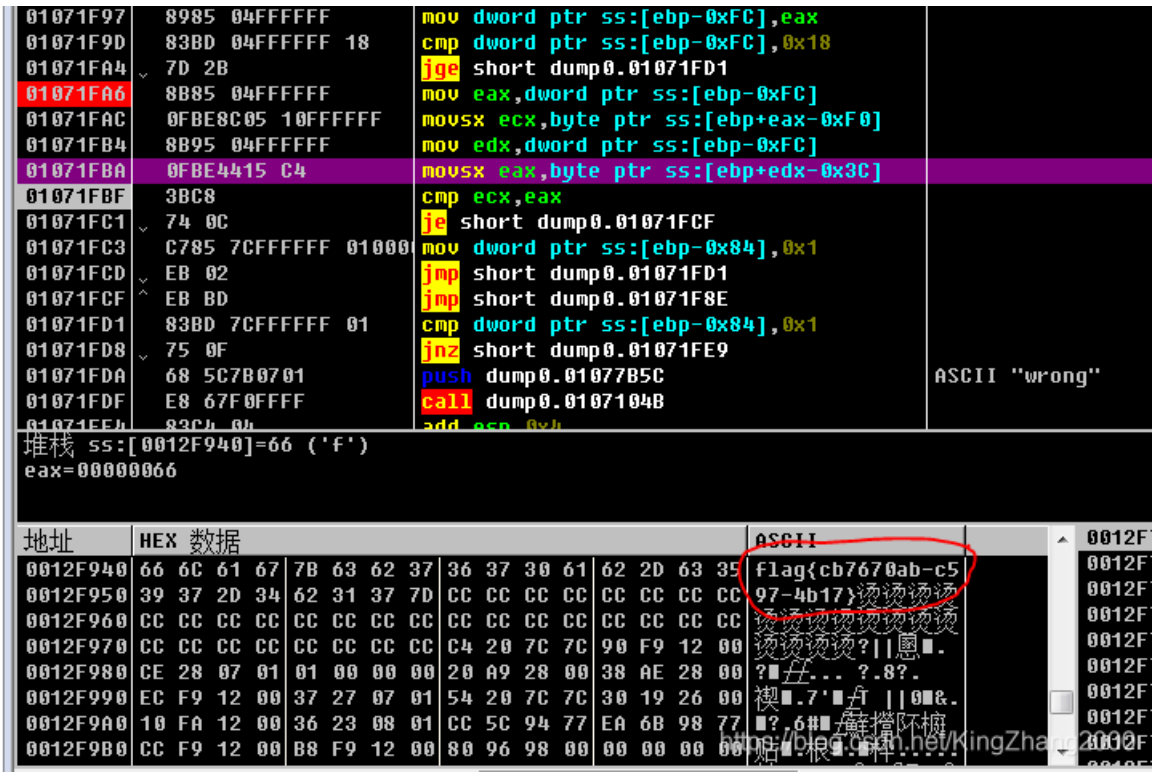
```
sub_10710BE(Buf);
for ( i = 0; i <= 23; ++i )
  v79[i] = *(&v29 + i) ^ *(&v53 + i) ^ Buf[i];
v4 = 54;
v5 = 108;
v6 = 106;
v7 = 104;
v8 = 44;
v9 = 33;
v10 = 59;
v11 = 58;
v12 = 43;
v13 = 38;
v14 = 112;
v15 = 37;
v16 = 105;
v17 = 42;
v18 = 97;
v19 = 61;
v20 = 83;
v21 = 99;
v22 = 52;
v23 = 35;
v24 = 112;
v25 = 116;
v26 = 42;
v27 = 37;
sub_10713BB();
for ( j = 0; j < 24; ++j )
{
  if ( *(&v4 + j) != v79[j] )
  {
    v77 = 1;
    break;
  }
}
if ( v77 == 1 )
  sub_107104B("wrong");
else
  sub_107104B("good");
sub_1071262(&savedregs, dword_1072028, 0, v0);
return sub_107123F((unsigned int)&savedregs ^ v80, v1);
}
```

buf 就是用户输入的字符串，if (*(&v4 + j) != v79[j]) 这个就是关键的比较，那么V79[]就应该是异或后的结果，也就是ben本题的密钥，于是在OD中定位值：6ljh,!;:+&p%i*a=Sc4#pt*%



接下来就简单了，直接把这个密钥输入，然后再次定位到这块就能得到flag了





最终得到flag：flag{cb7670ab-c597-4b17} 输入到shy.exe 验证一下：)